Bosco

Project Proposal

—

Margaret McCarthy
Student Number : W20095610
Higher Diploma in Science in Computer Science

Stakeholders :	ServisBOT Ltd.
Project Type :	Work Based Project
Project Category :	CI/CD Pipeline

Background

ServisBOT Ltd runs an online platform that creates chatbots which allows end users to communicate with a business or service through a pop up messenger on a website. The technology is cloud based and mostly serverless which means it is provided over the internet rather than using storage on a physical computer or server. Because of this it has to be constantly monitored and so tests are run continuously to ensure any problems are detected and resolved immediately. Testing the code is crucial to the service they provide. It ensures a robust platform and gives customers confidence that the system is dependable.

The components of a chatbot and each of its functions are broken down into microservices. These are created independent of one another but combined are the building blocks for the chatbot. There is a suite of tests which ServisBOT have called Frankenstein tests which run against these microservices. They run several times an hour, every hour. The tests use software called Testcafe which is an end to end testing service that uses messenger in a browser in order to run their tests. It simulates what a user would do by opening a browser and interacting with a chatbot. If the chatbot reacts in the expected way the test passes, otherwise the issue is investigated and resolved.

Project Overview

The Issue

All tests are run on two EC2 instances in two different AWS regions and as such it causes a lot of contention for CPU and memory. When a test run is instigated all tests are competing for CPU usage because each instance is running thirty plus tests on one server which has limited memory.

Also there are numerous problems with Testcafe. Testcafe has proven to be resource heavy, expensive and inefficient, causing slow CPU performance. The tests are difficult to maintain, because of the competing resources some tests affect the performance of other tests and would be more efficient if they ran in isolation. Debugging and monitoring of the tests is complex. The tests do not scale. Scalability is essential in order to increase the number of tests according to the number of user interactions and alternatively to scale down the tests when interactions drop which would not only prove to be more cost effective but would mostly solve all the issues mentioned.

The Proposed Solution

This project proposes to investigate whether it is possible to do a complete overhaul of the test suite by migrating the tests from the EC2 instances to Lambda functions. This will ensure they will all be run independently and are not competing for memory. It would mean the test suite could be scaled infinitely, would run faster and as a result would cost a lot less.

In particular the migration will focus on an automation tool called Puppeteer which has been proven by ServisBOT to be more performant than Testcafe. The Puppeteer package includes its own browser whereas Testcafe launches an external browser which is a slow process. With Puppeteer there is more control over what the developer can test, it is more efficient, easier to debug and has a lot more functionality. It is widely chosen by developers now.

Process

The purpose of this project is to find a solution to the current problematic architecture in place. The means by which this will be achieved is to carry out the following process.

1. Cost analysis of the current system in order to compare it to an alternative.
2. Investigate why the tests are slow, what are the weaknesses and if all else fails if they can be improved on.
3. Proof of concept. Run a singular test on a lambda using Puppeteer.
4. Ramp up the process by migrating more of the Frankenstein tests from ECS to Lambda using Puppeteer instead of Testcafe.
5. Analysis will be carried out on scalability, compatibility with different browsers, developer experience, efficiency and cost. The data gathered will give an insight into whether the overhaul is justifiable.

Technologies

Amazon Web Services (AWS)

ServisBOT runs many AWS services as it is a cloud based platform. Many of which will be used for the project including the following :

- Lambda functions and containers. Each test will be run on a lambda container.
- Cloudwatch Synthetics : these simulate customer interactions in order to test your code.
- Cloudwatch Metrics : To monitor data
- Codebuild : Part of the CI/CD process by rebuilding your code and running tests against it.
- ECS : virtual servers on which to run applications. Although the project aims to move away from ECS, a review of the current process which uses ECS will be carried out.
- S3 : S3 buckets are used for storage of large files
- Cloud Formation : template for creating AWS infrastructure. Eg: Lambda functions

Other Technologies

Docker Containers : Cloud based storage containers for Docker images

Programming Languages

Javascript

Markup Language

Tools

Puppeteer : An automation tool used for testing

NodeJS : Javascript runtime environment

Visual Studio Code : Code editor

APIs : A form of communication between two computer programs

GitHub : Version control for code and documentation

Slack : Used for the CICD process.